CHAPTER 4

Going Public!

Now that you have something to publish, you need to know how to get it to your Web server. Do you have a Web server? Do you have your own domain? What is your platform? In this chapter, we have a worksheet to help you get answers to the right questions.

Once you get your files up to your server, you want to organize them effectively so you always know how to link between files. We discuss the various approaches to organizing your files and weigh in with our opinion on the best way (as if we could resist). Finally, we discuss the methodology of testing your files. When you finish this chapter, you'll be ready to publish your files correctly, manage the directory structure, and test your site thoroughly.

In This Chapter

Desktop Web servers

Enterprise Web servers

Publishing on your own server

Publishing on your service provider's server

Publishing to an NT server

FTP clients

Developing directories

Uploading pages and graphics

DNS

Permissions

Testing your work

Getting Your Facts Straight

To publish your page in a way that enables visitors to your site to view it properly, you should know a few things. This section helps you ask the right questions of your system administrator, or ISP, if you have one, or to make the right decisions about deploying your own Web server, if you are thinking about that option.

We start with the worksheet. If you already have this information at your fingertips, you can skip the definitions of each item and go directly to the detailed instructions about publishing.

Information on the Web site publishing worksheet

The following worksheet is for you to fill out. Having this information in writing should be helpful, especially if you are doing something other than publishing for a month or so. You may well have forgotten how you did everything. You obviously don't need to write down your password. Many times a systems administrator creates a standard password for you and then you're expected to change it to something no one can guess. This would be the place to write the password the administrator gives you.

Web Site Publishing Worksheet

My Web server is:
___ on my desktop
___ not on my desktop

My Web server administrator is:
___ me
___ someone within my organization
___ someone outside my organization

My Web server platform is:
___ UNIX
___ Windows NT/2000
___ Mac

My desktop platform is:
___ Windows 95
___ Windows NT/2000
___ Mac
___ UNIX

My Web server name is:
DNS: _____
IP address: _____
NT domain: _____
NT name: _____
NT share name: _____

Contact information:
Name: _____
E-mail: _____
Phone: _____

My account information is:
Account ID: _____
Password: _____

Desktop Web Servers

Do you have your own Web server? If you hunt around on your computer, you may find you do have Web server software on your computer. The most common Web server software for your desktop is Personal Web Server by Microsoft, which is free from the Microsoft Web site, but others exist. You are most likely to have acquired one of these packages when downloading some Web-related software from a Web page.

You can install a desktop Web server, if you wish. Should you? Probably not. The disadvantages of using your desktop machine for a server outweigh the advantages. Consider the following pros and cons:

Pros	***Cons***
No charge for disk space usage. Most ISPs charge you by the megabyte you use, after some small initial amount included with your monthly fee. Many organizations have disk quotas enforced on their servers and getting more space is difficult or impossible. If you have a 2GB hard drive and you want to have 1GB of images on your Web site, you can do it free on your desktop. (We discuss why you don't really want 1GB of images in your Web site later.)	You can't ever turn off your machine. You have to think twice when you want to install new software and would need to restart your computer for the software to take effect. This is doubly true if you have any downloads from your site. Having a Web site going up and down all the time doesn't look professional.
You have control over the directory structure. Many of these pros are autonomy issues. This is certainly one. You may be working within an organization or with an ISP that has strict rules about where you can put HTML files, where you can put images, where you must put anything you want to execute (scripts), and where style sheets must be stored. If you can't live with those constraints, then you may need your own Web server.	You have to solve all your server problems yourself. Are you ready to start worrying about registry entries in Windows 95 or NT? Do you want to handle security issues from your desktop? The alternative is to turn your whole server administration over to someone else. Don't think of this as some big loss of autonomy. You don't learn how to dry-clean just so you can dry-clean a few suits and sweaters, do you? No. You are willing to depend on the dry cleaner to do it right because doing it yourself for the little you do simply isn't worth the trouble.
	You must have a high-speed direct connection to the Internet with a permanent IP address. A dial-in connection to the Internet is inadequate.

Continued

Pros	*Cons*
You can create as many mappings as you like. Consider the URL `http://www.overtheweb.com/purchase`. You know there is a server called `www.overtheweb.com`, but is there really a directory called *purchase?* It doesn't matter. Your Web server can create virtual directories, called *mappings,* that point visitors to some remote location on the Web server with one easy-to-remember virtual directory name. Some ISPs charge you for creating mappings. They are convenient for you as a developer—creating links between documents—and for visitors, too.	
You can associate as many server-side applications with your Web site as you like. If you want to run any server-side processes, such as forms processing or database interaction, you may want to use something that isn't built into your desktop Web server. Most ISPs charge for the software and again for setup. Some ISPs charge by the month for using specialized server-side applications. Some organizations won't install server-side applications on their Web servers for maintenance reasons. On your own computer, you can install whatever you want.	
Ease of publishing a page. This one applies to everyone. Publishing a page is as simple as saving it to the right directory on your hard drive.	

Enterprise Web Servers

The alternative to a desktop Web server is an enterprise Web server. Many varieties of these exist: O'Reilly Web Site Pro, iPlanet Web Server Enterprise Edition, Netscape Enterprise Server, Microsoft Internet Information Server, Apache, and many others. Generally, they run on UNIX or Windows NT, although they can also run on Macs.

An enterprise Web server can be within your own organization or with your Internet service provider. In either case, there is a systems administrator who should be able to answer all your questions.

An enterprise Web server is usually a robust, turnkey environment for your Web site. It is directly connected to the Internet (as opposed to being connected by a modem). Usually there are backup procedures and an uninterruptible power supply (UPS) to prevent downtime due to power outages.

Publishing on Your Own Server

If you have your own server, you should know just where to save your files. If your server is an enterprise server, you may have to transport them using FTP, but you probably know about all this. When you are publishing to your own server, it's easy to forget good practices of file organization. Heck, no one enforces any rules on you. For this reason, deciding on a directory structure and being disciplined about following your own rules is even more important. If you don't impose discipline on yourself, you can have a big mess on your hands in no time.

Skip down to the section called "Developing Directories to Store Your Pages and Graphics" to learn more about the philosophy choices you have. Make some rules for yourself and follow them.

When you are the systems administrator

For some lucky few, you will have your own Web server and it will reside on a dedicated server—then you are a systems administrator. You are the person everyone else must come to for answers to these questions. You are the top dog.

As the systems administrator, you can have all the hard drive space you want, all the mappings you want, and, as long as it doesn't conflict with other vital services your server provides, you can install all the additional applications you want.

Publishing on Your Service Provider's Server

Mere mortals must consign themselves to publishing their pages on someone else's server. The major disadvantage is loss of autonomy: You have to play by someone else's rules. However constraining this sounds, its advantages are overwhelming:

- **A functioning, reliable server provides a turnkey environment for your site.** You needn't pull anything out of a box, install any software, or find answers to difficult technical questions. You just sit at your desk, develop your site, and then when the time comes, you publish to your site and everything works (from the server side, anyway).
- **Someone else already has answers to your questions.** Even though your systems administrator may be hard to find, he or she does have answers to your questions. Heck, at one time, your systems administrator didn't know the answers and had to find them out by doing research. It is so much easier for you to be able to ask your systems administrator instead of digging out the answers yourself!
- **More time to create Web sites. What do systems administrators do?** They install software. They answer questions from their users. They create new accounts. They answer questions from their users. They perform backups and restores. They answer questions from their users. They create Web sites. Oh, yeah, and they answer questions from their users. You can spend your time developing great Web sites instead of all these other fun things.

As an industry, ISPs have grown out of nowhere. This isn't the place to discuss everything you need to consider before selecting an ISP, but know both great ones and awful ones exist. Ask for referrals. Let someone else keep an account at the awful one. If your ISP isn't great, switch. Even though they can't guarantee the network will always be up, your ISP should take great pains to keep you informed. This is the minimum you should demand.

Platform Issues

Now that you have found out who your systems administrator is, you can find out on what platform your Web server runs. How you move files between your desktop and your server depends on three things:

1. **Your desktop platform.** Are you sitting at a Mac, a PC, or a UNIX workstation, or at a terminal running X-Window? Even if you are at a PC or a Mac, if you operate within the UNIX environment, for example, from a telnet session, UNIX is your platform.

2. **Your server platform.** Ask that elusive systems administrator.
3. **The HTML editor you use.** We discuss this in depth in Chapter 8. The good HTML editors have built-in publishing capabilities. You can even save your file directly to a UNIX server from a PC or a Mac.

Because we don't yet know which HTML editor you'll like the best or even whether you want to fork out the dough to get one, and because your plain old text editor did such a nice job in the last chapter, we proceed assuming you don't have any fancy tools to get your pages to your server.

Extra: Publishing to a Windows NT/2000 server

Publishing to a Windows NT/2000 server is easier than publishing to a UNIX server. Unlike UNIX, an NT server can be mapped as a hard drive on your PC or on your Mac. This means, instead of using FTP, which we discuss next, you can copy your files as if you were just copying them from one directory (or folder) to another. This won't work if you are running Windows 3.1 on your desktop (but why would anyone still be running Windows 3.1?).

From Windows 98

You can connect your Windows 98 desktop computer to an NT server in two ways: the easy way and the other way. The easy way isn't available to everyone, but everyone can use the other way.

First, the easy way:

1. **Make sure you have Client for Microsoft Networks and NetBEUI running.** You can do this by
   a. From the Start Menu, select Settings and then Control Panel.
   b. Select the Network Control Panel.
   c. On the Configuration tab, make sure you have both Client for Microsoft Networks and NetBEUI listed, as shown in Figure 4-1.
2. **If you don't, you must add them and then restart your computer.** Client for Microsoft Networks is a *Client* and NetBEUI is a *protocol*. Chances are, if these weren't already running, you won't be able to use the easy way.

Figure 4-1: Network Control Panel in Windows 98

3. **Open your Network Neighborhood.** If you see your NT server on the list of available machines, you're in luck. Try checking under Entire Network. Otherwise, skip down to the other way. You still need to have accomplished Step 1.
4. **Double-click the NT server that is your Web server.** This opens a window with all the share names on the NT server. These are directories or disks available to be shared, if you have permissions.
5. **Right-click the share name you were instructed to use.** This brings up a list of the things you can do with the share.
6. **Select Map Network Drive.** This brings up a dialog box in which you can select the drive letter you want to associate with the server, as shown in Figure 4-2. You can actually map more than one drive from a single NT server; each would get a drive letter.

Figure 4-2: Mapping a network drive in Windows 98 is easy when your NT server is in your Network Neighborhood.

7. **Select the drive letter you want to associate with the NT server.**
8. **Check the Reconnect at Logon box, if you want to map to this drive each time you log in.** If you don't check this box, you must repeat Steps 2–6 each time you want to copy or save files to your NT server.

Now, you can either save your HTML files directly to your Web server directory or save them to your local hard drive, wherever you want to put them, and then copy them to your Web server directory when you are ready to publish. Graphics files, which can sometimes be tricky, move over correctly if you just copy them this way.

Now, the other way:

1. **Make sure you have Client for Microsoft Networks and NetBEUI running.** See Step 1 under the easy way for instructions.
2. **Edit your HOSTS and LMHOSTS files.** These can be found in your Windows directory. Both these files have no extensions. There may also be HOSTS.SAM and LMHOSTS.SAM files, but those are different files. Don't touch those! If you don't have either file, create them. If you create them using Notepad, it adds `.txt` to the end of the file name. You must find these files in your Windows directory and rename them without an extension.

   You need both the name, the domain name, and the IP address of your server. Your systems administrator can provide these for you. You can get the IP address from your systems administrator or, if he or she gives you the domain name (such as `nautilus.minerva.net`), you can go to a DOS prompt on any computer hooked up to the Internet and type **ping your.domain.name** and the system will return the IP address of `your.domain.name`. The domain name sometimes has four parts, as in `nmc.itc.virginia.edu`. Here is what your files need to have:

   LMHOSTS file:

   ```
   206.55.47.87 nautilus  #pre #dom:minerva
   ```

   HOSTS file:

   ```
   206.55.47.87 nautilus.minerva.net
   ```

   The second item in the LMHOSTS file is the NT name of the server. You have to get this from your systems administrator. The `#pre` tells your system to preload this network connection. The `#dom:` tells the system it should find the `minerva domain`. Again, you need to get the domain name from your systems administrator. In the HOSTS file, the item after the IP address is the server's domain name.
3. **Restart your computer.**

4. **Sign into the Microsoft Networking dialog box with the ID and password provided by your systems administrator** (even if this isn't what you normally do). You may get weird errors from your regular servers, if you normally sign into a server other than the NT Web server (such as a different NT server or a Netware server), but just click through them.
5. **Right-click Network Neighborhood.** You get a list of the things you can do with networking.
6. **Select Map Network Drive.** This brings up the dialog box, shown in Figure 4-3, where you can indicate what you want to map (your NT server) to which drive letter.

Figure 4-3: Mapping a network drive to a remote host in Windows 98

7. **Enter the NT server name, followed by the share name in the first blank as follows:**

   ```
   \\nautilus\users
   ```

   Again, you don't actually type `nautilus\users`; you type your machine name, followed by the share name given to you by your systems administrator. Be sure to type backslashes (\), not forward slashes (/), or it won't work.
8. **Enter the drive letter to which you want to map in the second blank.** This will default to the next available letter of the alphabet.
9. **Check the Reconnect at Logon box, if you want to map to this drive every time you log in.** If you don't check this box, you must repeat Steps 4–8 every time you want to copy or save files to your NT server.

From Windows NT/2000

You can attach a Windows NT/2000 workstation to a Windows NT/2000 server in three ways. For all these methods using Windows NT, you must have Client for Microsoft Networks and NetBEUI running. If your server is Windows 2000, you can use IP, and NetBEUI is no longer needed. You can see which protocols are running by checking the Network Control Panel, under the Services tab. If these are already running, try the easy way explained in the previous section on Windows 98. If these aren't already running, you need administrator privileges on your NT workstation to install these services. If you can't use the easy way, use this other method:

1. Right-click Network Neighborhood. This brings up a list of system options.

2. Select Map Network Drive. A dialog box like the one in Figure 4-4 pops up.
3. Enter the following in the first blank:

   ```
   \\206.55.47.87\users
   ```

   The first part is your IP address and the second part is your share name. Make sure you use backslashes (\) or it definitely won't work.

Figure 4-4: Mapping a network drive using an IP address in Windows NT

4. Enter your login ID in the second blank.
5. Indicate the drive letter you want to assign.
6. Check the Reconnect at Logon box.

Now you can save or copy files to the drive letter you indicated. If you click My Computer (or whatever you have renamed your computer), you see a list of all your drives, including the one you just mapped to your NT server.

From a Mac

The only way to map an NT server as a local drive on a Mac is if you are sharing a network and the NT server has Services for Macintoshes turned on. If this is the case and you can find the NT server in the right zone in the Chooser, then you can sign on to the NT server, using the share name the systems administrator gave you. Then you can save or copy files directly to your NT server directory.

If you save or copy HTML or CSS files from your Mac to the NT server and then you try to open them from a PC, they will look strange, but they will still work properly. Figure 4-5 shows a file with a line-length problem.

If you save or copy from a PC and then open on a Mac that has a mapping to the NT server, then your file will look like the one in Figure 4-6.

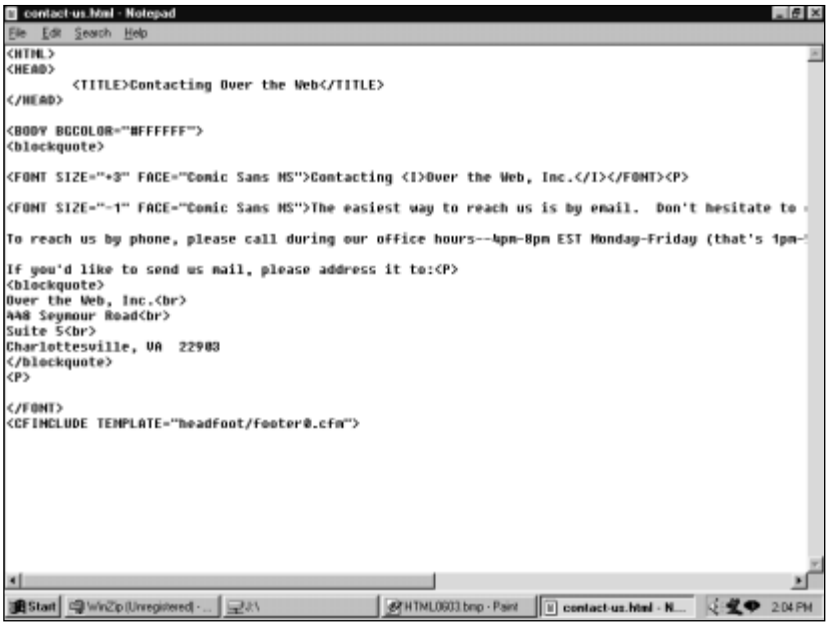

Figure 4-5: This is what your HTML or CSS file looks like if saved or copied from a Mac and then opened on a PC.

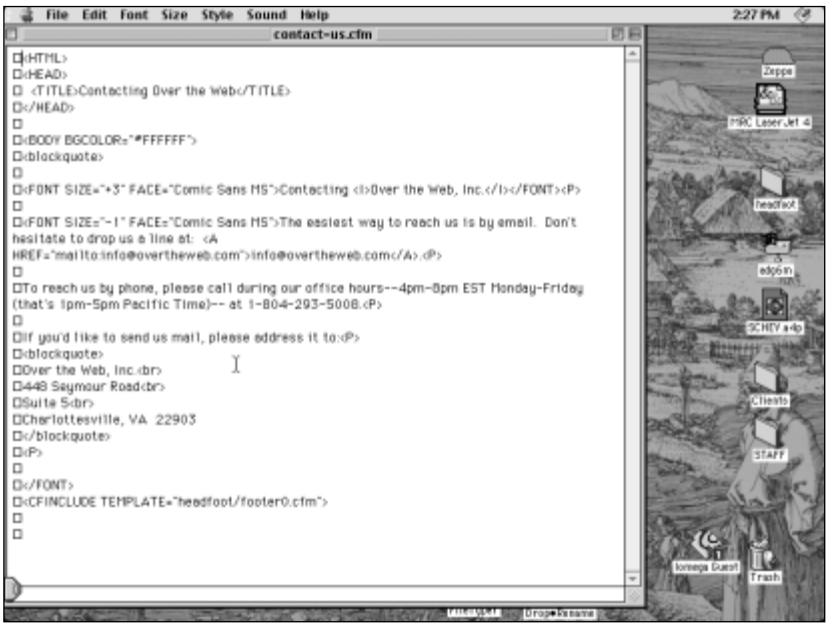

Figure 4-6: This is what your text files look like if opened on a Mac, after having been saved or copied from a PC.

Introducing FTP

The other way to move files from your desktop to your Web server — if you don't have an NT server or you can't connect to it for some reason — is the tried and true method of transferring your files, called File Transfer Protocol (FTP). FTP is a fast, easy way to move your files from any desktop to any server.

In the early days of FTP and still today in non-X-Window environments on UNIX, all file transfers relied on a text-based interface. Commands were short and sweet, but only one file could be moved at a time. This was the entire vocabulary: open, get, put, bini (for binary file transfer), and bye. Figure 4-7 shows an example of FTP commands.

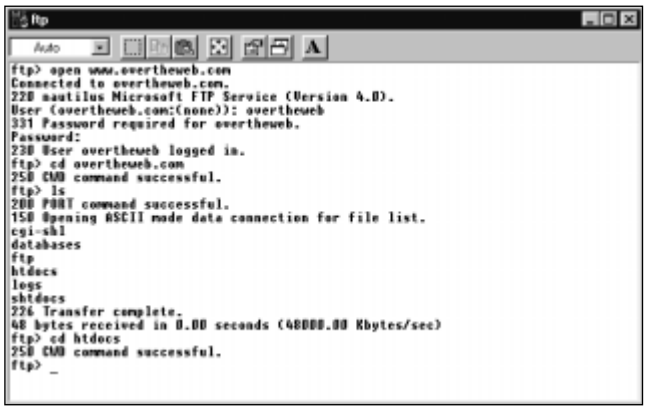

Figure 4-7: Text-based FTP client

You will upload two kinds of files: binary and text. When you think about it, everything on your computer is binary, even your text files. *Binary* just refers to the way the data is stored: as ones and zeros. With a *text file,* every set of 8 ones and zeros translates neatly into a letter, a number, or a special character on your keyboard. With a binary file, all the ones and zeros translate into something bigger, such as an image. If you open a text file with a plain text editor, you see text. If you open a binary file in a plain text editor, you see junk. The plain text editor has tried to turn every set of 8 ones and zeros into a number or letter.

FTP predates the Web

Before there was a World Wide Web, there was an Internet. The World Wide Web relies on the HyperText Transfer Protocol (HTTP). The Internet relied on the File Transfer Protocol (FTP). FTP is a simple concept. One computer establishes a connection with another computer using TCP/IP (more on this follows). Then the first computer tells the second computer whether the data about to be passed will be text or binary data (*binary data* is any file that looks like gobbledygook when opened in a plain text editor; this would include word processing documents, spreadsheets, and image files). Then the first computer either puts one file onto the second computer or gets one file off the second computer. All this assumes the first computer has the necessary permissions to read or write files on the second computer. That's all: so simple, so elegant.

FTP takes advantage, as does HTTP, of TCP/IP. This is a double-decker protocol. The Transmission Control Protocol (TCP) part packages the data a certain way so when the packages get to the other side, the computer over there knows how to unpackage them and reassemble the data into the right file. When the receiving computer gets data, it needs to know whether it is putting together an e-mail message (plain text) or an image. The Internet Protocol (IP) part handles getting those packets to the correct address. TCP is like the mailroom and IP is like the post office.

Introducing FTP clients: CuteFTP, WS_FTP, Fetch

Once every computer had a mouse, drag-and-drop FTP software was not long in coming. Today, several visual FTP clients are available. For the Mac, it is easy: Fetch. Fetch even guesses whether your file is text or binary (what it calls *raw data*). You can drag a file right out of the folder on your desk into the Fetch client and watch the dog run. Figure 4-8 shows Fetch capabilities on the Mac.

For the Windows environments, the most common are WS_FTP, for which a lite version is available, and CuteFTP (shown in Figure 4-9), which is shareware. In both of these packages, you can set up a list of the most common places to which you connect, including your login ID, your password, and the directory on the remote machine. Then you can drag-and-drop files or directories full of files from your desktop machine (by pointing the software to the right directory) to the Web server (or any remote machine that has an IP address). This software makes a reasonably accurate guess as to whether your file is binary or text.

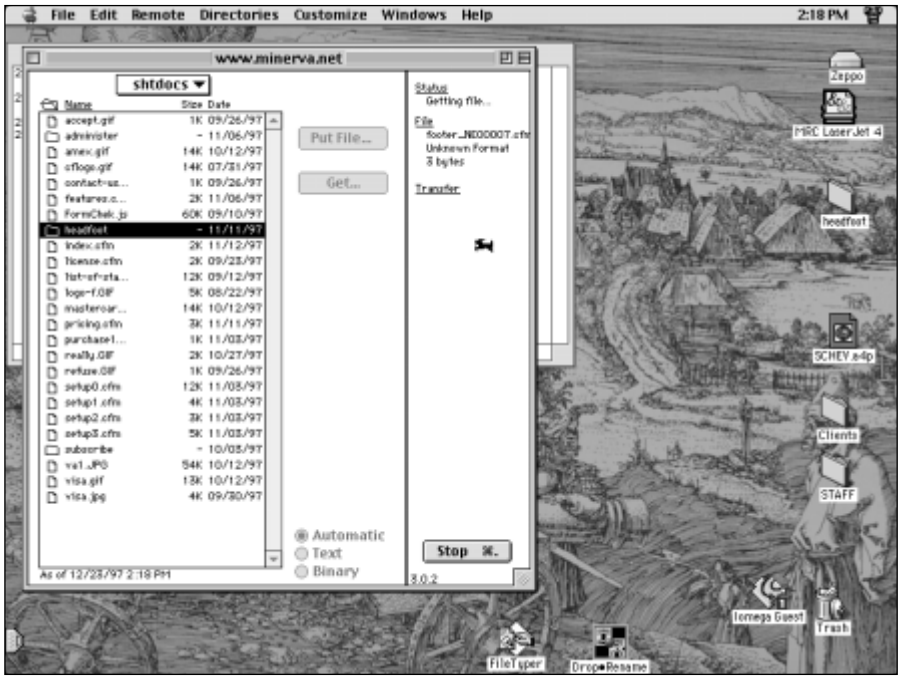

Figure 4-8: Fetch makes FTPing easy on the Mac.

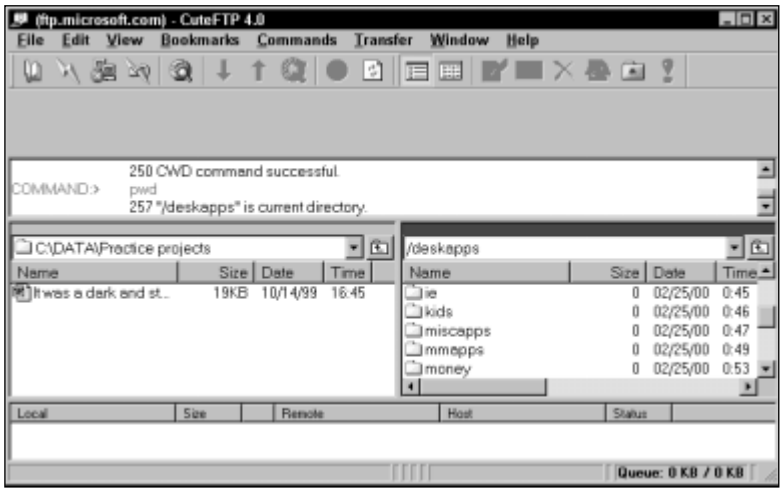
(ftp.microsoft.com) - CuteFTP 4.0
File Edit View Bookmarks Commands Transfer Window Help
250 CWD command successful.
COMMAND:> pwd
257 "/deskapps" is current directory.
C:\DATA\Practice projects
/deskapps
Name Size Date Time
It was a dark and st.. 19KB 10/14/99 16:45
ie 0 02/25/00 0:45
kids 0 02/25/00 0:46
miscapps 0 02/25/00 0:47
mmapps 0 02/25/00 0:49
money 0 02/25/00 0:53
Local Size Remote Host Status
Queue: 0 KB / 0 KB

Figure 4-9: CuteFTP makes light work of FTPing files and folders.

Developing Directories to Store Your Pages and Graphics

Before you start uploading those pages, you should decide how you want to structure your directories. You have three choices: put everything into one directory, put your files into a directory structure that mirrors your site (for example, one directory per page), or put your files into directories by type. Making this decision early and enforcing discipline on yourself to keep your structure clean is important. The alternative is to recode all your links when you rearrange your files.

Some of the tools to make this easier are found in Chapter 8—but it is still not much fun.

Unless your site is small (one page with a photo of your cat and your grandmother's baklava recipe), you don't want to put all your files into one directory. This leaves two choices: having your directory structure match your pages, which we tell you in the next section heading is generally a bad idea, or structuring your directories by file type, which is, generally, a better way.

The reason we include an explanation of both theories is, for some organizations, having your directory structure come closer to matching your pages than to being by file type does make sense. This is the case in a large organization when dozens of people publish pages to one site. You probably don't want everyone to have permission to overlay everyone else's files. Breaking up the directory structure by functional group enables you to protect each group's files from carelessness by anyone in any of the other groups (say, two people naming their files the same thing).

Directory structure to match your pages

The idea behind this is pretty straightforward. For each page, you create a directory. You call it something related to what's on the page. You can either use a flat directory structure, where every page is on the same level, or a hierarchical directory structure, where a page that serves as a pointer to four other pages is actually closer to the root than those four pages, which are under it. Figure 4-10 shows a directory structure that matches the pages in the site.

Having one directory per page doesn't make sense, but it may make sense to have one directory per functional unit involved in creating your site. You could modify the previous diagram by removing the directories for features (a subset of sales), FAQs (a subset of support), and purchase (a subset of sales), and lumping those three in with the pages they modify. If you have a handful of people working on each unit, you can protect your site better by restricting the permissions of the people in each group only to the folder where all their files go. Figure 4-11 shows an even more convoluted directory tree with one page per directory using hierarchical structure.

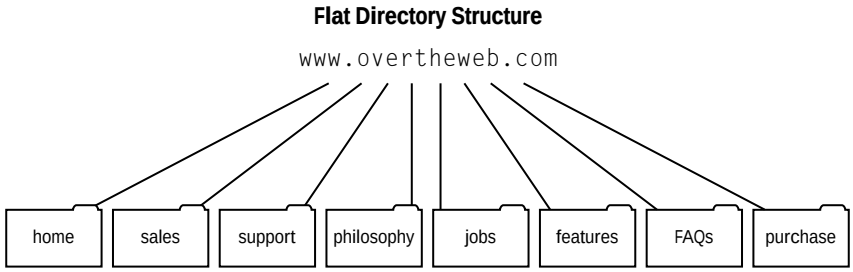

Figure 4-10: Flat directory structure with one directory per page

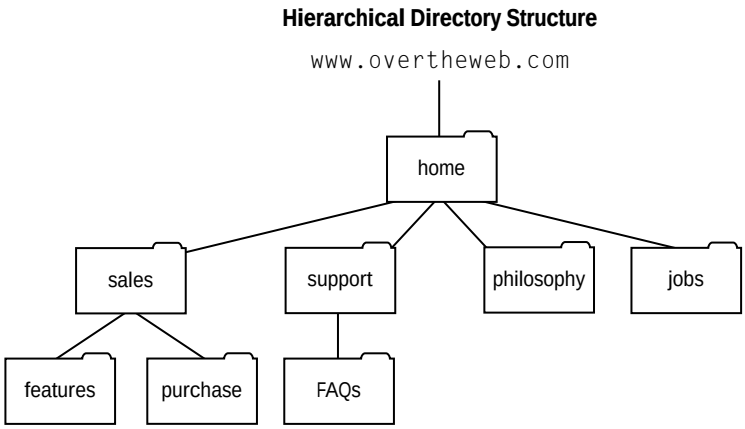

Figure 4-11: Hierarchical directory structure with one directory per page

It's hard to imagine where this model makes sense because it just doesn't scale. If you rearrange your site, you may as well start over.

Directory structure by file type (generally a better idea)

HTML files go into the root directory for your site. Style-sheets files go into a style-sheets directory (as shown in Figure 4-12). All media go into subdirectories of a media directory based on media type (if you think you'll have more than just images — be realistic). If you think you'll have only images, skip the media directory and have a directory in your root directory named images.

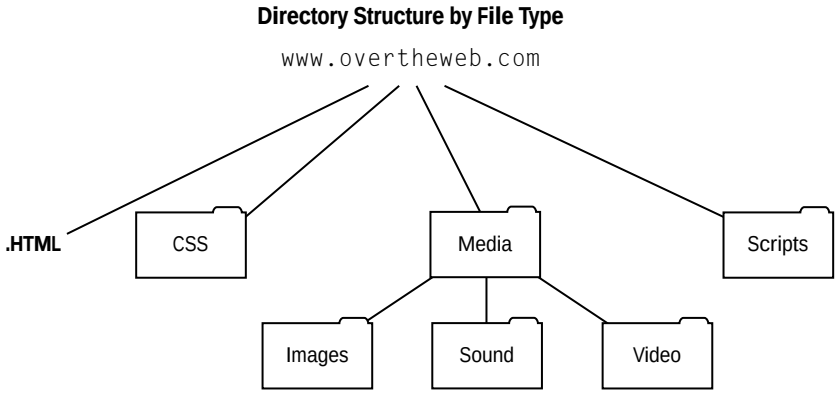

Figure 4-12: Nice and simple. Directory structure by file type. Especially good for small sites.

Uploading Your Pages and Graphics

Earlier in this chapter, we mentioned text files are binary files where every set of 8 ones and zeros neatly convert into one character. This is almost perfectly true. Text files are more than what you see. If text files were only text, each file would be one long line of text. In fact, at the end of each line in a text file is at least one character indicating this is where a line ends.

PCs use two characters: carriage return (CR) and line feed (LF). Macs use only CR and UNIX uses only LF. What a pain! Indeed it is, but when you FTP your files, your FTP client can take care of this for you. If you tell your FTP client (or let it guess) your file is a text file, then it automatically makes the proper adjustments to the end of the lines to correct for the file's new platform. If you tell your FTP client your text file is actually binary, then it leaves the whole file alone. This means when you open your file on the new platform, the line ends won't be right.

What if you upload your binary file as text? Disaster. The FTP client has conveniently gone through your file and changed every string of 8 ones and zeros that translate into an LF or CR and tried to convert them as if they were text. You won't recognize your file and neither will a browser.

If you are working on either a Mac or a PC and your Web server is UNIX, or if you are working on UNIX or a Mac and your Web server is Windows NT, then try this experiment: upload a text file (any old HTML file will do) to your Web server as text and then download it as binary. Your end of file characters will be wrong because you didn't let the FTP client fix it on the way back down.

DNS

What will you call your site? Do you have a domain name (like `ibm.com`)? You can do this from the official organization that registers domain names, called InterNIC: `www.internic.net`. You can also get your ISP to do it for you, but they usually charge something extra. One enterprising group on the Web chose to call itself `www.internic.com`. Guess what it does? It registers your domain name with the official group and charges you twice as much! The group has easy online forms, but usually your ISP will do it for less than twice the registration amount.

Before you can get a domain name, you must have the technical information from your ISP, such as the IP address of your ISP. This is yet another good reason to let your ISP register your domain name for you.

Permissions

In addition to needing an account on your Web server, you also need permissions to perform basic file management tasks. You must be able to create new directories within your root directory, unless your systems administrator is a real dictator and creates them for you according to a prescribed system. You need permissions to write to your directory to upload files.

When visitors come to your site with their browsers, they either read or execute files on your Web server. The permissions for the directories must be set to permit this. Your systems administrator will probably want all your executable files (scripts and programs) in one directory so just that directory can be marked as executable.

Testing Your Work

You have to test your work. Because of the informal system of creating pages and publishing them, which most of us work in, this is easy to forget. Don't wait until you receive mail from someone who visited your site telling you links don't work to learn this. Your visitors shouldn't be your beta testers.

Where should you test? On your server. While this sounds dangerous, you can ask your systems administrator to create an obscure mapping name, so if a browser comes to `www.overtheweb.com`, it returns a 404 not found error while you are testing your site. You will know the only way to see the site is to go to `www.overtheweb.com/blahblah`. When you are done testing, you ask your systems administrator to delete the `blahblah` mapping and reinstate `www.overtheweb.com` to point to your home page.

What about testing on a desktop server? Good reasons exist not to do it this way. The following chart discusses the pros and cons of testing on a desktop Web server, if this is not where you ultimately plan to publish.

Pros	*Cons*
Permissions are not an issue because you should have God-like permissions on your own workstation. The exception to this is if you are running Windows NT Workstation or Windows 2000 Professional on your computer. If you are running NT and you don't have administrator privileges, then you may not even be able to install a desktop Web server.	Permissions may be different on your production Web server. If your permissions aren't the same on your desktop server as they are on your production Web server, then you need to make changes. How will you know they're different? You have to test. This means you will test your page twice.
Testing is instant. Just save your HTML file, as we did in the last chapter, and reload your page.	Testing is less thorough. Why? Because you know you have to test again anyway. You'll be better off using the mapping suggestion we made previously and doing your testing once on your production server.
No special file manipulation is required. Just save your pages to your Web server's home directory, which is right on your local hard drive.	Unless you are doing server-side scripting, you get the same results as if you just opened the file locally. Let me explain. In Chapter 3, we taught you how to open a local file from your browser. If you install a desktop Web server, you can pull up the file using the IP address of your computer, if you know it, or `http://127.0.0.1/yourpage.html`, if you put your page into your desktop Web server's root directory. Instead of opening the file locally, it opens the file as if it were coming from somewhere on the Internet. Your page looks exactly the same as it did by opening it locally. The only time this kind of round-about page loading is necessary is if you are doing some kind of server processing of your page. In this case, you would also have some other software on your desktop computer (such as Cold Fusion, Tango, FileMaker Pro, and so on).

The most important points to remember when you test are as follows:

- ✦ Look at your site, clicking through all the links from a variety of browsers (both Netscape and Internet Explorer in their most recently released versions and one or two versions earlier). Looking at your site from a few versions of AOL's browser is also not a bad idea because they have such a large installed base.
- ✦ Look at your site from both Macs and PCs.
- ✦ Look at your site after setting your screen resolution to 640×480, 800×600, and 1024×768.
- ✦ Look at your site with the browser sized to the full screen and with the browser taking up only part of the screen.
- ✦ Look at your site from within your domain and then from outside your domain. Permissions may be different if you are coming to your site from outside your domain.
- ✦ Look at your site from the desk of someone who doesn't have privileges on your server.

Both Windows NT/2000 and Windows 98 send your login data to an NT/2000 Web server (pretty tricky, huh?), so if you are authorized to see a certain page because of your ID and password, the Web server already knows this and doesn't even bother to ask you to sign in. If you are not authorized, you get a sign-in screen. Of course, as the Webmaster, you never see this screen—thus, you never know everyone else can't get to your site—unless you try this from someone's computer who doesn't have the permissions you have. In fact, if you didn't know this, you'd never be able to duplicate this permissions problem from your own desk and you'd assume everyone coming to your page was doing something wrong, because it worked for you!

Complete a testing worksheet, something like the one shown in Figure 4-13, and you won't go wrong.

From Here

Find servers fascinating? Wish you could be a systems administrator and have your own enterprise Web server? Then jump to Chapter 11 to learn more about server options.

Proceed to Chapter 5 and get your feet wet with an introduction to HTML.

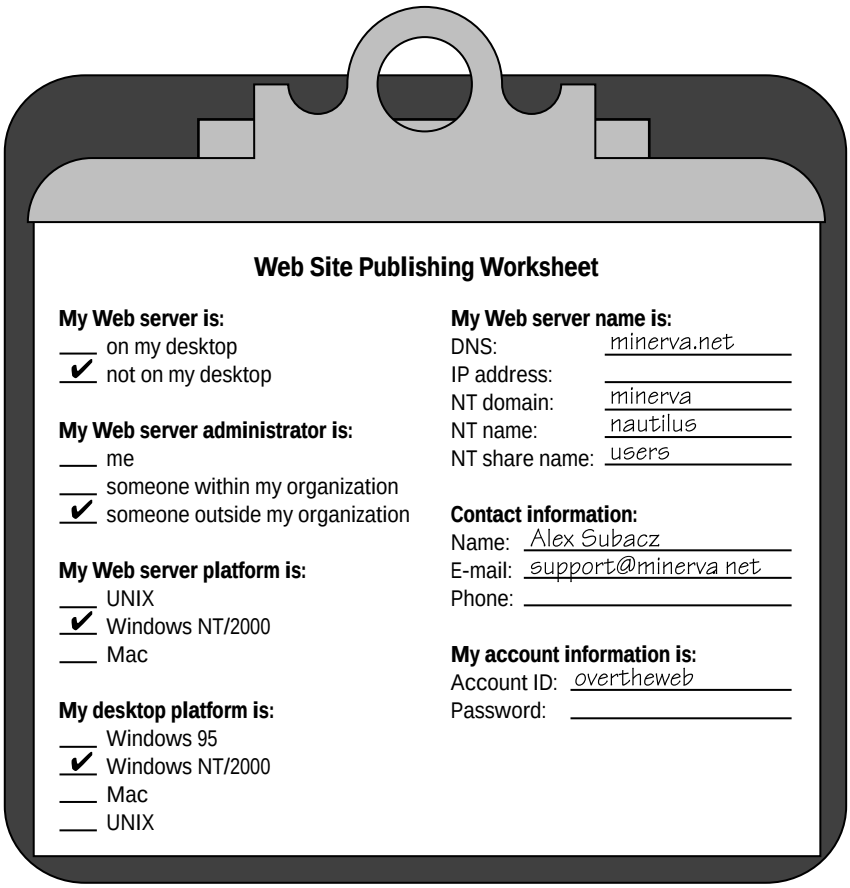

Figure 4-13: This is what your worksheet may look like once you've completed it.

Summary

You now have everything you need to begin publishing your pages. You have a completed worksheet with all the information you need about your server. You understand file types. You have made some decisions about how you want to structure your directories. You know how to get your files to your server using an FTP client. Or, if you have an NT Web server, you know how to map a drive to your NT server.

You're ready to publish your files correctly, manage the directory structure, and test your site thoroughly.

✦ ✦ ✦